Defining Rules for Kinematic Shapes with Variable Spatial Relations

Conference or Workshop Item

Published as
Harrison L., Jowers I., Earl C. (2019) Defining Rules for Kinematic Shapes with Variable Spatial Relations. In: Lee JH. (eds) Computer-Aided Architectural Design. "Hello, Culture". CAAD Futures 2019. CCIS vol 1028. Springer, Singapore

Defining Rules for Kinematic Shapes with Variable Spatial Relations

Laura Harrison[1], Iestyn Jowers[2], Chris Earl[3]

The Open University, Milton Keynes, UK
[1] laura.harrison@open.ac.uk
[2] iestyn.jowers@open.ac.uk
[3] c.f.earl@open.ac.uk

Abstract. Designing mechanisms can be a challenging problem, because the underlying kinematics involved are typically not intuitively incorporated into common techniques for design representation. Kinematic shapes and kinematic grammars build on the shape grammar and making grammar formalisms to enable a visually intuitive approach to model and explore mechanisms. With reference to the lower kinematic pairs this paper introduces kinematic shapes. These are connected shapes with parts which have variable spatial relations that account for the relative motion of the parts. The paper considers how such shapes can be defined, the role of elements shared by connected parts, and the motions that result. It also considers how kinematic shape rules can be employed to generate and explore the motion of mechanisms.

Keywords: Shape grammars, kinematic design, making grammars, boundaries.

1 Introduction

In shape grammars, abstract shapes model the pictorial representations used during design activities [1]. The shape grammar formalism is well suited to visual explorations of these representations, and the computational mechanism of shape rules has been applied to describe and support creative design processes [2]. This is because the shapes used in shape grammars are visually dynamic, supporting reinterpretation and recognition of emergent forms. In recent years, shape grammars have been extended to *making grammars* [3], where the aim is to formalise physical manipulations of material and objects, and to represent processes of making that take place in arts, crafts, and manufacturing. Consideration of *things* made from *stuff*, introduces new constraints to ensure that shapes mimic the behaviour of physical objects in physical space, for example to take account of collisions [4]. This paper is concerned with a subclass of physical objects, mechanisms with moving parts [5], and

Final authenticated version is online at https://doi.org/10.1007/978-981-13-8410-3_31

explores the constraints that arise when shapes are used to represent and explore mechanisms, in *kinematic grammars*.

A variety of well-proven methods exist for designing mechanisms, e.g. [6], but the underlying kinematics involved are typically not intuitively incorporated into common techniques for design representation. In some instances, linked static representations (such as series of images) may communicate the combined effects of the possible motions of parts within a design. Alternatively, physical or virtual models can be used to test motion – either through simulation or material interaction. But in general, exploration depends on a designer's ability to apply understanding of potential motions between parts to independently predict and model (mentally or otherwise) the combined effects within a designed object.

Building on shape grammars, kinematic grammars aim to provide a formalism which will enable a visually intuitive approach to model and explore mechanisms [5]. In abstract terms, the motion of mechanisms can be modelled according to connected objects that move relative to each other. Consequently, kinematic grammars incorporate shapes with explicit but connected parts that have variable spatial relations between them. In this paper, kinematic grammars are introduced with reference to a specific class of mechanisms, the *lower kinematic pairs*. The paper proceeds in the next section by introducing the lower kinematic pairs; Section 3 considers the concept of *kinematic shapes* as models of physical mechanisms; Section 4 explores how mechanisms can be explored using kinematic grammars; and, Section 5 discusses kinematic grammars with reference to lower kinematic pairs.

2 Mechanisms in Motion

At its most basic, the design of a mechanism can be described according to combinations of the relative motions of connected parts [8]. The pairs of parts that give rise to motions are often referred to as kinematic pairs, and are subject to certain spatial conditions. Firstly, one of the parts needs to be fixed with respect to the local spatial neighbourhood. Which of the two parts is fixed is of no consequence because the motion is relative, and temporally the part only needs to be fixed for the duration of the motion. Secondly, the geometry of the two parts needs to restrict the relative motion in some way. The result of this is that the fixed part determines an envelope of motion for the moving part. In order to ensure motion, the shared geometry of the connected parts must have the same curvature. This means that the shared geometry of parts must either be a point, or it must have constant curvature, i.e. it is either rectilinear, circular, or a helical combination.

Kinematic pairs are classified in various ways: according to types of connection, i.e. surface, line or point; according to the type of relative motion, e.g. sliding or rolling; or according to the type of constraint applied to the pair, e.g. mechanical or due to gravity. Here, the focus is on a particular classification of kinematic pairs, referred to as lower pairs. These are identified according to a surface connection, and are differentiated from higher pairs, where connection is a point or a line, e.g. the connection between a cam and its follower. In total, there are six lower pairs, as

illustrated in Fig. 1. The lower pairs enumerate spatial restrictions on motion, resulting in pairs of parts with relative motions of varying degrees of freedom (DoF):

- *Prismatic pair* (slider), e.g. Fig. 1i: the axes of the two parts are aligned, allowing translation along the axes and no rotation. This results in one DoF
- *Revolute pair* (hinged joint), e.g. Fig. 1ii: the axes of the two parts are aligned, allowing rotation about the axes and no translation. This results in one DoF
- *Screw pair*, e.g. Fig. 1iii: the axes of the two parts are aligned, allowing a combination of translation and rotation relative to the axes. This results in one DoF
- *Cylindrical pair*, e.g. Fig. 1iv: the axes of the two parts are aligned, allowing independent translation and rotation relative to the axes. This results in two DoF
- *Spherical pair* (ball joint), e.g. Fig. 1v: the spherical centres of the two parts are aligned, allowing rotation about three axes and no translation. This results in three DoF
- *Planar pair*, e.g. Fig. 1vi: the surfaces of the two parts are in contact, allowing translation in two directions and rotation about one axis, perpendicular to the surfaces in contact. This results in three DoF

In the design of a mechanism, kinematic pairs can be combined in chains to create models of complicated motions [6]. These are often abstracted as graphs or hypergraphs of links and nodes which can be used to determine the potential motion of a mechanism, based on connections, but without consideration of geometry [7]. Consequently, when a design is realised as a physical model, complications can arise when geometry interacts or collides during the motion of parts.

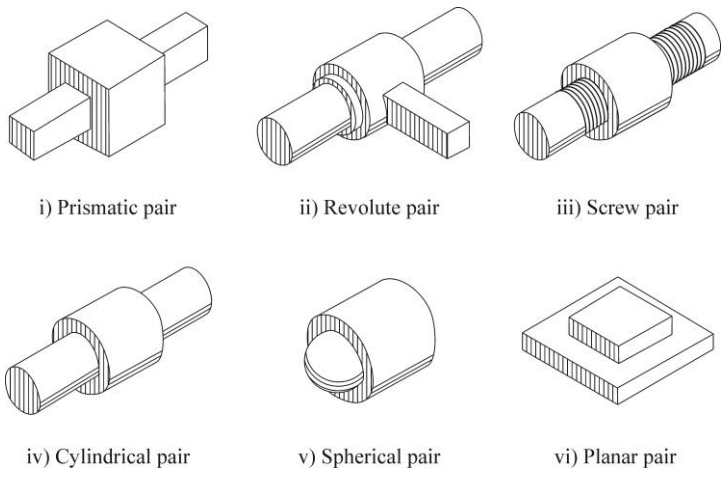

Fig. 1. Examples of the six lower kinematic pairs

The spatial nature of kinematic pairs implies that mechanisms can be readily described as shapes in shape computations, and there are certain benefits in doing so. Shapes can provide a model of a mechanism that includes geometry as well as

connection of parts, whilst retaining a level of abstraction that can support creative exploration. Designers are primarily concerned with modelling physically realisable designs. However, real motions are not necessarily easily described using shape computation. Conversely, more abstract notions of motion which could not be achieved in the physical world can give interesting results when modelled virtually, and therefore should not be precluded from investigation. Shape computations can be used to design and explore mechanisms in a designer-friendly way which is visually intuitive [5]. In this paper, kinematic shapes are used to model mechanisms and their motion, with reference to the lower kinematic pairs.

3 Kinematic Shapes

In a shape grammar, shape rules are used to generate designs through consideration of shapes and the spatial relation between shapes and/or parts of shapes [1]. Any two shapes (or parts of a shape) define a spatial relation. For example, all of the shapes in Fig. 2 are composed of the same three parts: a small square, a larger square, and a point located at their shared vertex. But, the shapes are all distinct from each other because of the different spatial relation between the two squares. Shape grammars often make use of such relations through applications of shape rules which produce repetition of form and arrangement and can result in visually cohesive patterns, or designs consistent with a particular style [9].

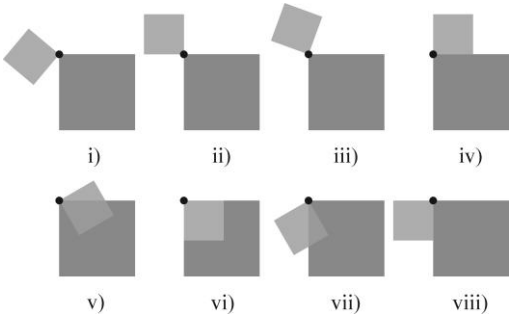

Fig. 2. Examples of spatial relations

The spatial relations used in a shape grammar are typically fixed, or for parametric shape grammars are instantiated during application of a shape rule. Spatial relations can change via application of shape rules but this does not reflect the behaviour of mechanisms, such as the lower pairs illustrated in Fig. 1, where spatial relations between parts change continuously according to their motion. Therefore, to support formal exploration of mechanisms via shape computation, it is necessary to consider the motion according to *variable spatial relations* (VSR) between parts, i.e. the continuously changing relation between parts that are in motion.

VSRs result from well-defined motions of parts, and a *kinematic shape* is a shape which includes one or more VSRs between its parts. For example, in the eight shapes

in Fig. 2 the spatial relations between the small and large squares vary according to the rotation of the small square about the point. These eight shapes can be recognised as instantiations of a kinematic shape in which there is a VSR between the two squares, defined according to a rotation of the small square. This motion is not a consequence of transformations realised during application of shape rules, it is instead an implicit property of the kinematic shape. As kinematic shapes, all eight of the shapes in Fig. 2 are equal, and comparison with Fig. 1 reveals that they are a two-dimensional equivalent of a revolute pair (Fig. 1ii). This example highlights the key features of kinematic shapes; they include connected parts that are in motion.

3.1 Shapes in Motion

Shape algebras [1] provide a framework suitable for exploring motion of shapes, as summarised in Table 1. The algebras are denoted U_{ij}, where i is the dimension of the shape elements used to construct a shape, j is the dimension of the embedding space, and $i \leq j$. Motion of a shape is defined according to a reference shape which is a shape element of dimension k, where $k < j$, and the lower dimensional embedding spaces, defined by points and lines, are more restrictive with respect to motion than the higher dimensional spaces of planes and volumes.

Table 1. Shape motion in algebras U_{ij}

Algebra	Space	Motion	Reference	DoF
U_{00}	Point	-	-	-
U_{i1}	Line	Translation	Point	1
U_{i2}	Plane	Rotation	Point	1
		Translation	Point	2
		Translation	Line	1
U_{i3}	Volume	Rotation	Point	3
		Rotation	Line	1
		Translation	Point	3
		Translation	Line	1
		Translation	Plane	2

In the algebra U_{00}, the embedding space is a single point, and no motion is possible. While in algebras U_{i1}, the embedding space is a straight line, shapes are composed of points or lines, and the only possible motion is translation. This is defined relative to a point, with one degree of freedom (DoF). Algebras U_{i2} are familiar to designers who work with sketches to develop design concepts. The embedding space is a plane, shapes are composed of points, lines or planes, and motion is composed of rotations and translations. Rotation is defined relative to a point, with one DoF, and translation is defined relative to a point, with two DoFs, or relative to a line with one DoF. Algebras U_{i3} are analogous to physical space, or the 3D space within a CAD system. The embedding space is a volume, shapes are composed of volumes, planes, lines or points, and as with U_{i2} motion is composed of translations and rotations. Rotation is defined relative to a point, with three DoFs, or a

line, with one DoF, while translation is defined relative to a point, with three DoFs, or a plane, with two DoFs. As an example, Fig. 3 illustrates moving shapes in the algebra U_{22}, where shapes composed of planes are arranged in a plane. Representing motion in a static image can be difficult, and Fig. 3 adopts a convention of using arrows to indicate the motion of the squares. In Fig. 3i, a square is rotated around a reference point, in Fig. 3ii, a square is translated relative to a reference point, and in Fig. 3iii a square is translated relative to a reference line.

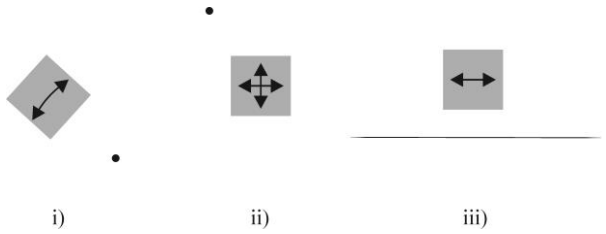

Fig. 3. Moving shapes in U_{22}

Shapes with moving parts, i.e. kinematic shapes, can also be formalised in algebras U_{ij}, by considering relative motions of the parts that results in a VSR. The simplest kinematic shapes are described by a triple of shapes, $\{s, \alpha, e\}$, where s represents a static part, α represents a moving part, and e represents a reference shape, as enumerated in Table 1. The existence of multiple parts which move independently distinguishes the kinematic shape from a shape in motion, but the motion of α, the moving part, is not defined relative to s, the static part. For example, the kinematic shape in Fig. 2 is composed of two squares, and the rotation of the small square is not defined relative to the large square, it is instead defined relative to the point, which acts as the reference shape. In general, the motion of a moving part α is defined relative to e, the reference shape, which is a shape element, of dimension k, in an algebra U_{ij}, $k < j$. The VSR therefore defines the spatial relationship between α and e, and $VSR(\alpha, e)$ is a shape given by an instantiation of the motion of α relative to e. A simple kinematic shape is therefore given by $s + VSR(\alpha, e)$. For connected kinematic shapes, such as the shape illustrated in Fig. 2, the VSR can be determined by considering the connectivity of the parts s and α.

3.2 Shapes with Connected Parts

Shapes are connected when they touch, and a shape is said to be a *connected shape* when each part touches some other part [1]. For example, Fig. 4 illustrates different connected shapes in U_{22}, composed of two squares, labelled x and y. In Fig. 4a, the squares are connected because x is a subshape of y; in Fig. 4b, they are connected because they overlap; and in Fig. 4c-f they are connected because they touch, either at their edges or at their vertices.

Shape connectivity can be defined in terms of the recursive embedding relation applied to parts, boundaries of parts, boundaries of the boundaries of parts, etc. [1]. The boundary of a shape in an algebra U_{ij} is a shape in an algebra $U_{(i-1)j}$, and the

operator $b^i(S)$ formalises this recursive relation between boundaries b, and shapes S, with integer $i \geq 0$ and $b^0(S) = S$. For example, a shape S in U_{33} is composed of volume shape elements and has a boundary $b(S)$ composed of planes in U_{23}. This in turn has a boundary $b^2(S)$ composed of lines in U_{13}, which in turn has boundary $b^3(S)$ composed of points in U_{13}. For all the connected shapes in Fig. 4, x and y contain parts that share a boundary, i.e. an edge, or a boundary of a boundary, i.e. a vertex.

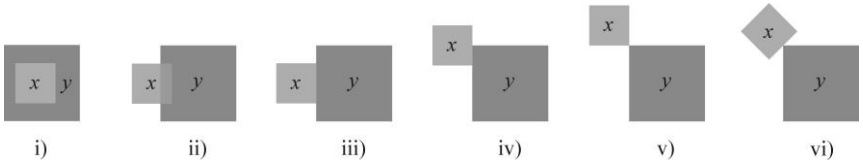

Fig. 4. Examples of connected shapes in U_{22}

Generalising this example, using the boundary operator and subshape relation $\leq$, two shapes, x and y, can be defined as connected if there are shapes z and z' such that $z \leq b^i(x)$ and $z' \leq b^j(y)$, and $b^k(z).b^l(z')$ is not the empty shape, with integers $i, j, k, l \geq 0$. This definition can be applied to the connected shapes in U_{22} illustrated in Fig. 4, as follows, although there may be many possible choices of z and z' in each case:

- in Fig. 4i, x is embedded in y; z and z' are both in U_{22}, $z \leq x$ and $z' \leq y$ so that $z.z'$ is not the empty shape
- in Fig. 4ii, x and y overlap; z and z' are both in U_{22}, $z \leq x$ and $z' \leq y$ so that $z.z'$ is not the empty shape
- in Fig. 4iii, x and y share part of their boundary; z and z' are both in U_{12}, $z \leq b(x)$ and $z' \leq b(y)$ so that $z.z'$ is not the empty shape
- in Fig. 4iv, x and y share part of their boundary; z and z' are both in U_{12}, $z \leq b(x)$ and $z' \leq b(y)$ so that $z.z'$ is not the empty shape
- in Fig. 4v, x and y share a vertex; z and z' are both in U_{02}, $z \leq b^2(x)$ and $z' \leq b^2(y)$ so that $z.z'$ is not the empty shape
- in Fig. 4vi, an edge of x touches a vertex of y; z is in U_{12}, z' is in U_{01}, $z \leq b(x)$ and $z' \leq b^2(y)$ so that $b(z)$. $b^0(z')$ is not the empty shape

This definition of shape connectivity is fairly intuitive, and captures the idea that shapes are connected if their parts touch. It also applies to shapes in composite algebras, which are composed of spatial elements of different dimensions.

In shape grammars, connectivity between parts of a shape is temporary and changing, depending on the application of rules that dynamically alter the structure of a shape [1]. Similarly, in kinematic shapes, the connectivity between parts also changes, but not according to rule applications, instead according to different instantiations of the VSR, given by $VSR(\alpha, e)$. For example, in Fig. 2 as the small square rotates about the point, the connectivity of the two squares changes: in Fig. 2i-iii, the two squares are connected due to the shared vertex, but in Fig. 2iv&viii, they are connected due to a shared boundary, while in Fig. 2v&vii, they are connected due to a shared part, and in Fig. 2vi, they are connected because the small square is embedded in the large square. If they are retained, these different connections have

implications with respect to the potential motion of the small square, as illustrated in Fig. 5. In these examples, connectivity of the small and large squares is explicitly identified by the shape elements drawn in black, and the arrows are used to indicate the motion of the small square according to the VSR. Fig. 5i combines all the shapes from Fig. 2 into a single kinematic shape; the two squares are connected at a shared vertex and the VSR is defined by the rotation of the small square about this point. The shape is a U_{22} equivalent of a revolute pair (Fig. 1ii). In Fig. 5ii, the two squares are connected at a shared edge, and the VSR is defined by the horizontal translation of the small square parallel to the edge. The shape is a U_{22} equivalent of a planar pair (Fig. 1vi). In Fig. 5iii, the two squares overlap and are connected by a shared subshape. Consequently, they are locked in position and the small square cannot move. In these three examples, the spatial relations of the two squares are instantiations of the kinematic shape illustrated in Fig. 2, but different interpretations of the connectivity of the kinematic shape give rise to different possible motions. Ambiguity about how connectivity of shape is interpreted can be reduced by explicitly including connecting shape elements as part of the shape; for example these are drawn in black in Fig. 5.

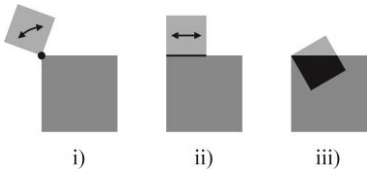

Fig. 5. Motion of connected shapes

In these examples, the connecting elements also act as a reference shape, and define the motion of the small square, and this is potentially of benefit. When the connecting shape element and the reference shape are different a restriction of potential motion can result, as illustrated in Fig. 6. In Fig. 6i, rotation of the small square about the reference point, identified at its centre, is restricted due to the connectivity of the two squares, as specified by the black line on the shared boundary. The connectivity of the two squares is such that only vertical translation of the small square is possible, as illustrated in Fig. 6ii, where the connecting shape is the reference shape. Alternatively, if a rotating part is required, then this issue can be resolved by changing the geometry of the parts, as illustrated in Fig. 6iii where the small square is replaced with a circle. The boundary of a circle is invariant under rotation, and as a result the specified motion is not restricted by the connectivity of the parts. The shape in Fig. 6iii is a U_{22} equivalent of a spherical pair (Fig. 1v).

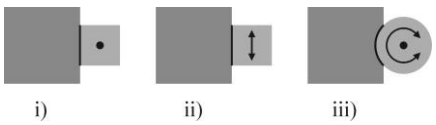

Fig. 6. Exploring the consequences of shape connectivity

The explicit inclusion of connecting shapes means that kinematic shapes begin to behave more as physical objects, where motion is defined not only according to abstract concepts such as reference shapes, but also according to interactions between parts. However, this gives rise to a conflict between the expected behaviour of shapes in a shape grammar, and the expected behaviour of a mechanism. For example, in Fig. 2, as the small square rotates it overlaps the large square so that there are parts of both that occupy the same region of the embedding space. Since shape grammars are a visual formalism, when this situation arises, it is common for overlapping shapes to merge and form a single shape element. However, for physical mechanisms it not possible for parts to occupy the same region of space, and parts in motion should remain distinct. This issue can be resolved by recognising that physical mechanisms behave as things made of stuff, and should be modelled within making grammars [5].

3.3 Kinematic Shapes as Things made of Stuff

Making grammars [3] apply the computational framework of shape grammars to physical objects, and model processes of making that take place in the arts, crafts, and manufacturing. To support this, shape algebras are extended to include spatial *stuff*, which is the composite matter of physical *things*, and making grammars incorporate actions applied to stuff as consequences of sensing its properties, e.g. by seeing or touching. Examples include knotting of strings in Incan khipu and painting with watercolours [3]. However, as discussed in the previous section, physical objects exhibit different behaviours to shapes, and these should be taken into consideration when exploring how computations for making grammars might work within shape algebras. Krstic [4] identified key factors that distinguish things and stuff used in making grammars from shapes and parts used in shape grammars. These are concerned with the equivalence of representations in different algebras, and the treatment of boundaries.

Stuff and things are by definition three-dimensional, and in shape algebras are therefore most naturally represented in algebras U_{33}, where volumes are arranged in three-dimensional space. For the purposes of illustration, it is also useful to consider two-dimensional equivalents in algebras U_{22}, where planes are arranged in two-dimensional space. In design, it is common to use boundaries as a representation for a shape, e.g. in solid modelling surface-based models are common, with U_{33} objects represented in U_{23}. However, representing material things for making in an algebra U_{ij}, where $i < j$ can give rise to conceptual inconsistencies. This is illustrated in Fig. 7 where a shape rule $a \rightarrow b$ is represented in three different algebras, which are visually similar but conceptually very different.

Fig. 7. Examples of shape rules in i) U_{12}, ii) U_{22} and iii) $U_{22} \times U_{12}$

In the rule in Fig. 7i, the shapes a and b are both in U_{12} and this is a typical example of a rule from a shape grammar for design applications. It is an addition rule, with a square (composed of four lines) added to the square recognised by the left-hand side, and the partial order of the shapes in the rule is $a < b$. Compare this to the rule in Fig. 7ii, where the shapes a and b are both in U_{22} and the rule could be from a making grammar, e.g. to model cutting a hole in a sheet of material. It is a subtraction rule, with a planar square subtracted from the square recognised by the left-hand side, and the partial order of the shapes in the rule is $a > b$. These two rules are related by the boundary function $b(S)$, with the shapes in the rule in Fig. 7i being the boundaries of the shapes in the rule in Fig. 7ii. Because of this, and despite the visual similarity of the two rules, they perform opposite functions, one adds a square, while the other subtracts a square [4]. This example illustrates that the logic of rules for boundaries, e.g. the U_{12} rule in Fig. 7i, does not reflect the logic of making, which in this example is better modelled by the U_{22} rule in Fig. 7ii. But, rules for shapes in a U_{ii} algebra (such as U_{22}), where shapes and the embedding space are of the same dimension, are also problematic, because they are unconstrained, and can be applied to any shape in U_{ii} in infinitely many ways. Because of this, Krstic [4] suggests that in making grammars rules should include shapes and their boundaries, as illustrated in Fig. 7iii. This rule includes shapes from the composite algebra $U_{22}\times U_{12}$, and works as expected for a making grammar because the inclusion of boundaries provides context to ensure that the rule is applied correctly. Shapes in U_{ii} algebras together with their boundaries form a subalgebra of $U_{ii}\times U_{(i\text{-}1)i}$, denoted UB_i, which contains ordered pairs of shapes and their boundaries [11]. These algebras are weaker than shape algebras U_{ij}, because they lack Boolean operations of sum, product and difference, and partial order is defined component-wise, for shapes and their boundaries. But they are useful for modelling things in making grammars because they preserve the boundaries of shapes, which are useful to streamline rule application [4].

The UB_i algebras are closed under symmetric difference which can be used in the application of shape rules in a shape grammar. The symmetric difference of two shapes, x and y, is composed of the parts that are in either of the shapes, but not in their intersection and is given by $x \oplus y = (x - y) + (y - x)$, or equivalently $x \oplus y = (x + y) - (x \cdot y)$. For the boundaries of shapes in U_{ii} algebras symmetric difference is distributive, so that $b(x \oplus y) = b(x) \oplus b(y)$, where x and y are in U_{ii}, and for shapes in U_{ij}, it can replace sum when shapes are disjoint or difference when one shape is embedded in the other [10]. Specifically, $x \oplus y = (x + y)$ if $(x \cdot y) = 0$, and $x \oplus y = y - x$ if $x \leq y$. Using symmetric difference, a shape rule $a \rightarrow b$ can be applied to shape c under a transformation t when $a \leq c$ and $[c - t(a)] \cdot t(b) = 0$, to give $c' = [c \oplus t(a)] \oplus t(b)$ [4]. The subshape condition, $a \leq c$, ensures that the first instance of the symmetric difference results in a subtraction of $t(a)$ from c while the discrete condition, $[c - t(a)] \cdot t(b) = 0$, ensures that the second instance results in the addition of $t(b)$ to $c - t(a)$. A further condition on the boundaries of the shapes, $b(t(a)) \cdot b(c) \neq 0$, can be applied to provide registration for transformation t and restrict the applications of rules.

The discrete condition $[c - t(a)] \cdot t(b) = 0$ has the additional benefit of giving shapes the behaviour of physical objects during rule application, by avoiding collisions between parts. It ensures that the shape b on the right-hand side of the rule

does not collide with the shape that remains after subtracting the shape a from the left-hand side of the rule. For kinematic shapes this mechanism for collision protection is useful, and should be applied continuously to moving parts. To achieve this, a VSR condition should also be included, so that that in a simple kinematic shape composed of a triple of shapes, $\{s, \alpha, e\}$ the static part s and the moving part α should always be discrete, i.e. $s \cdot VSR(\alpha, e) = 0$. This will ensure that the parts of a kinematic shape do not overlap as a result of its motion.

4 Kinematic Rules for Kinematic Grammars

Inclusion of kinematic shapes in shape/making grammars requires a mechanism for distinguishing between parts of shapes that are in motion and those that are static. For this purpose, one of two opposing philosophical approaches can be adopted. It can either be assumed that, by default, parts are in motion, or that they are static. In the first approach, all parts of a kinematic shape are free to move around the embedding space, except for parts which are explicitly identified as being static. In the second approach, all parts of a kinematic shape are static relative to the embedding space, except for parts which are explicitly identified as being in motion relative to specified reference shapes. In terms of physical intuition, either approach is equally valid, since it can be recognised that, in general, objects tend to be free to move, unless they are constrained in some way while, when designing mechanisms, it is common to assume that parts are static unless their motion is specified. In this paper, the second approach has been assumed, and the notation used in the examples presented identifies parts of a kinematic shape that are in motion. Symbolically, the moving parts are represented with a Greek letter, while visually they are represented using a lighter shade of grey and are labelled with an arrow to indicate the resulting motion. This notation is simplistic, and useful for the exploration presented here, but there is perhaps benefit in exploring alternative representations using colour grammars [12] or weights [13] to support greater exploration of languages of kinematic designs.

In a kinematic grammar, motion can be introduced to a static shape by applying kinematic shape rules which take the form $a \rightarrow b + VSR(\alpha, e)$. Here, a and b are static shapes in UB_i, α is a moving shape, also in UB_i. e is shape element in U_{ki} ($k < i$) and acts as both a connecting shape element for b and α, and as a reference shape for the motion of α. $VSR(\alpha, e)$ is a UB_i shape given by an instantiation of the motion of α relative to e. Fig. 8 illustrates an example of a kinematic shape rule (Fig. 8i) and its application to a static shape (Fig. 8ii) to produce a kinematic shape (Fig. 8iii).

Fig. 8. Example of a kinematic shape rule and its application to a static shape

In Fig. 8iii the motion of the kinematic shape is illustrated by the inclusion of overlapping instantiations of the moving part, all of which observe the collision protection condition $s \cdot VSR(\alpha, e) = 0$. The result is a shape that is a two-dimensional equivalent of a revolute pair (Fig. 1ii). In this example, the logic of rule application follows the shape grammar formalism, and since $a \cdot b \neq 0$ the rule proceeds by recognising and replacing the shape $a - b$ with $VSR(\alpha, e)$. Alternatively, if $a \cdot b = 0$ then the rule would proceed by replacing a with b and adding a moving part.

Kinematic shapes can also be combined into chains, to model mechanisms with more complicated motions. In a kinematic grammar this can be achieved by applying kinematic shape rules which take the form $VSR(\alpha, e) \rightarrow VSR(\beta, e) + VSR(\gamma, f)$. Here, α, β and γ are moving shapes in UB_i. e is a shape element in U_{ki} ($k < i$) and is the reference shape for the motion of α and β. f is both a connecting shape element for β and γ and is also the reference shape for the motion of β. $VSR(\alpha, e)$, $VSR(\beta, e)$ and $VSR(\gamma, f)$ are UB_i shapes given by instantiations of the motion of the moving shapes α, β and γ relative to e, e and f, respectively. Fig. 9 illustrates an example of a kinematic shape rule (Fig. 9i) and its application to the kinematic shape in Fig. 8iii. In this example $\alpha \cdot \beta = 0$ and the rule proceeds by adding the second moving part, modelled by $VSR(\gamma, f)$. Alternatively, if $\beta < \alpha$, then $VSR(\alpha, e)$ is replaced with two moving parts modelled by $VSR(\beta, e)$ and $VSR(\gamma, f)$. As a result of applying the rule, the kinematic shape in Fig. 9ii has two parts in motion, both of which rotate about a connecting point. The result is a shape that is a two-dimensional equivalent of two revolute pairs (Fig. 1ii) combined in sequence. The motion of the kinematic shape is too complicated to be illustrated according to the method used in Fig. 8. Instead, in Fig. 9iii it is illustrated according to the envelopes of motion of the two moving parts. These define a sub-space of the embedding space and are represented as shaded regions. For both moving parts, the motion is restricted according to the connectivity of the parts, and according to the collision protection conditions $s \cdot VSR(\beta, e) = 0$, and $VSR(\beta, e) \cdot VSR(\gamma, f) = 0$, where s is the stationary part of the shape, and β and γ are the moving parts.

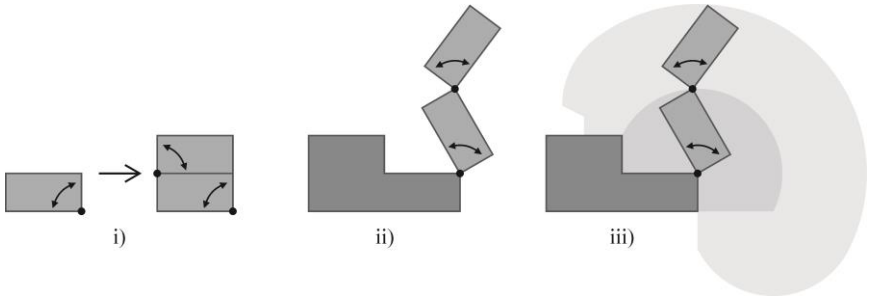

Fig. 9. Example of a kinematic shape rule and its application to a kinematic shape

Application of kinematic shape rules requires a mechanism for recognising embedded parts of a shape. This is complicated by VSRs, since spatial relations between moving parts are dynamic and cannot be used to provide registration for determining where shape rules can be applied. For example, the kinematic shape rule

illustrated in Fig. 10i is of the form $a + VSR(\alpha, e) \rightarrow b$ and its application requires recognition of both a static part and a moving part. Rules of this form can be used to remove moving parts from a kinematic shape, and include static shapes a and b, both in UB_i, moving shape α, also in UB_i, shape element e in U_{ki} ($k < i$) and $VSR(\alpha, e)$, a UB_i shape given by an instantiation of the motion of α relative to e. In applying the rule, recognition of a, the static shape on the left-hand side of the rule, follows the logic of rule application from shape grammars, where rule $a \rightarrow b$ applied to a shape c proceeds by first identifying a transformation t such that $t(a)$ is an embedded part of c, $t(a) \leq c$. The rule is then applied by removing the transformed instance of the shape a and replacing it with a similarly transformed instance of the shape b. In practice, identification of the transformation t is implemented by considering how distinct elements of a, such as vertices, are transformed and ensuring that all distinct elements of $t(a)$ are embedded in c [14]. For the moving shape on the left-hand side of the rule, this approach does not work, because the spatial relation between its distinct elements is changing according to the motion of α relative to e, and an alternative approach must be identified for recognising the moving parts of a shape. For this purpose, the invariants of the motion can be employed to identify the VSR between distinct elements, and the reference of motion. For example, to apply the rule in Fig. 10i to the kinematic shape in Fig. 10ii requires that the static and moving parts of the shape on the left-hand side of the rule are recognised as parts. Fig. 10iii illustrates the distinct elements of the shape in Fig. 10ii that are used to support this recognition. The motion of the moving part is a rotation about the connecting point, and consequently the distance from this point is invariant. In Fig. 10iii this is illustrated by dashed lines, which are of equal length for the two instantiations of the motion. These provide enough information to determine how the shape rule can be applied, and the result is the shape in Fig. 8ii.

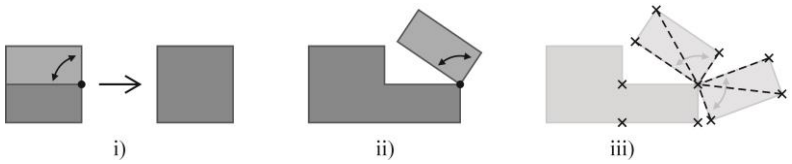

Fig. 10. Recognising parts of a kinematic shape

5 Discussion

This paper has explored how the shape grammar formalism can be applied to the problem of designing mechanisms with moving parts. With reference to the lower kinematic pairs (Fig. 1), kinematic shapes were introduced as connected shapes composed of static parts, moving parts and shape elements used as reference for motion. In essence each kinematic shape represents an infinite number of static shapes, each of which is given by an instantiation of the moving parts. Despite this it is still possible to recognise kinematic shapes and their moving parts for the purpose of rule application in a kinematic grammar by considering the invariants of motion.

A variety of kinematic shapes were introduced as illustrations, and these were identified to be two-dimensional equivalents of the revolute pair (Fig. 5i), the spherical pair (Fig. 6iii) and the planar pair (Fig. 6ii). The other lower kinematic pairs, the prismatic pair, the screw pair and the cylindrical pair, do not have two-dimensional equivalents, because they would violate the collision condition, which ensures that parts of shapes do not occupy the same region of an embedding space. For visual shapes this is common, e.g. overlapping planes, intersecting volumes, etc., but as models of mechanisms, kinematic shapes should behave as physical things composed of spatial stuff, and collision between moving parts should be avoided. However, the prismatic pair, the screw pair and the cylindrical pair can be represented as three-dimensional shapes, in an UB_3 algebra, as illustrated in Fig. 11. For each of these kinematic shapes, the moving cuboid is connected to the static shape by a shared surface, and the motion of the cuboid is with reference to a line that is parallel to this surface. If the collision condition is adhered to, then motion is defined and constrained both by the reference line and by the geometry of the static and moving parts. As a result, in the prismatic pair only translation parallel to the line is possible, in the screw pair, only a screw rotation with dependent motions about and parallel to the line is possible, and in the cylindrical pair two independent motions are possible, rotation about the line and translation parallel to the line.

To be of use in the design of mechanisms, kinematic grammars should give some indication of the resulting behaviour of kinematic shapes, i.e. the extent of the motion of the moving parts. Representing motion in a static image can be difficult, but in this paper various approaches have been employed to give some insight, including the use of arrows, Fig. 5, inclusion of multiple instantiations of moving parts, Fig. 8iii, and inclusion of envelopes of motion of moving parts, Fig. 9iii. Envelopes of motion are perhaps the most expressive of these, and as regions of space can themselves be modelled and analysed using shape arithmetic.

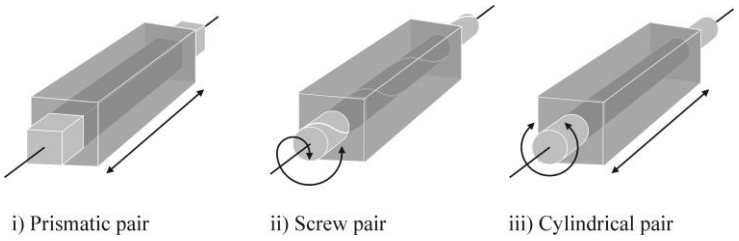

Fig. 11. Modelling kinematic pairs in U_{33}

The use of the lower kinematic pairs as a reference for defining kinematic shapes has resulted in certain restrictions. Only the motions of connected shapes have been considered, and as a consequence not all the motions enumerated in Table 1 are applicable. For example, in a two-dimensional embedding space, the moving parts of a connected kinematic shape can rotate about a point or translate parallel to a line, but cannot translate according to a point. A more general consideration of kinematic shapes could take into account all of the possible motions. Also, kinematic grammars have been identified as a variation of making grammars, where shapes adhere to

physical constraints, but with the inclusion of VSRs to account for moving parts. As a result, kinematic grammars do not readily support the visual emergence that typifies shape grammar applications, and much of the richness of the shape formalism has been lost. This is perhaps true for all making grammars since it is not obvious how visual emergence can work in a U_{33} algebra, when from any given view-point only part of a shape is visible, and only the boundaries (i.e. surfaces) of a part can be seen. However, there is still scope for reinterpretation of shape structure via shape rule applications, and emergence can arise in material behaviours of shapes in motion [15]. Kinematic connections between parts permit relative motions, and kinematic behaviours (or motions) emerge, as indicated in Fig. 9iii; also, when material deformations permit relative motions, elastic, plastic or even auxetic behaviours may emerge. Consequently rules acting to add multiple interacting VSRs within designs, can give rise to emergent kinematic behaviours that may be complex and surprising.

Future work will explore how kinematic shapes can be included in shape computation to formalise motions of physical materials and objects. A more general treatment could include broader categories of mechanisms than those considered here. Of particular interest is whether a visual approach can result in interesting designs, previously only found through application of analytical techniques.

References

1. Stiny, G.: Shape: Talking about Seeing and Doing, MIT Press, Cambridge, (2006)
2. Prats, M., Lim, S., Jowers, I., Garner, S., Chase, S.: Transforming shape in design: Observations from studies of sketching, Design Studies, 30(5), 503-520, (2009)
3. Knight, T., Stiny, G.: Making grammars: From computing with shapes to computing with things, Design Studies, 41, Part A, 8-28, (2015)
4. Krstic, D.: Grammars for Making Revisited, in J. S. Gero (ed.), Design Computing and Cognition ’18, Springer, 519-538 (2019)
5. Harrison, L., Earl, C., Eckert, C.: Exploratory making: Shape, structure and motion. Design Studies, 41, Part A, 51–78, (2015)
6. Tsai, L.-W.: Mechanism Design: Enumeration of Kinematic Structures According to Function, CRC Press, Boca Raton, (2000)
7. Berge, C.: Graphs and Hypergraphs, North Holland, Amsterdam, 1973
8. Reuleaux, F.: The Kinematics of Machinery. Outlines of a Theory of Machines (trans. and edited by A. B. W. Kennedy). Macmillan and Co, London, (1876)
9. Prats, M., Earl, C., Garner, S., Jowers, I.: Shape exploration of designs in a style: Toward generation of product designs, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 20(3), 201-215 (2006)
10. Earl, C. F.: Shape Boundaries, Environment and Planning B: Planning and Design, 24(5), 669-687, (1997)
11. Krstic, D.: Algebras and Grammars for Shapes and their Boundaries. Environment and Planning B: Planning and Design, 28(1), 151–162, (2001)
12. Knight, T. W.: Color Grammars: Designing with Lines and Colors, Environment and Planning B: Planning and Design, 16(4), 417-449, (1989)
13. Stiny, G.: Weights, Environment and Planning B: Planning and Design, 19(4), 413-430, (1992)

14. Krishnamurti, R.: The arithmetic of shapes, Environment and Planning B: Planning and Design, 7(4), 463-484, (1980)
15. Gürsoy, B., Özkar, M.: Visualizing making: Shapes, materials, and actions, Design Studies, 41, Part A, 29-50, (2015)